Modelo de cajas

El modelo de caja o (box-model) es el que condiciona el diseño de las páginas web.

En HTML, todos los elementos generan una caja que los contiene, sobre la cual, actúan los estilos.

Lógicamente las particularidades de cada caja varían en función del tipo de elemento que sea. Entonces, cada vez que se inserta una etiqueta HTML, se crea una nueva caja rectangular que encierra los contenidos de ese elemento.

Los navegadores crean y colocan las cajas de forma automática, pero CSS permite modificar todas sus características. Cada una de las cajas está formada por cuatro partes.

De adentro hacia fuera, las cajas tienen contenido, padding, borde y margen.

Contenido (content): Es el contenido HTML del elemento, como los parrafos, las listas, las imágenes, etc.

Relleno (padding): Es el espacio libre (opcional) que existe entre el contenido y el borde.

Borde (border): Es la línea que encierra el contenido y su relleno.

Margen (margin): Es el espacio libre (opcional) entre la caja y el resto de las cajas adyacentes.

Alto y ancho

Por default, si no le indicamos ningún ancho específico, el navegador va a hacer que la caja ocupe todo el ancho de la ventana. Con respecto al alto, va a hacer que la caja ocupe el mínimo posible, por lo que el alto de la misma, va a depender pura y exclusivamente del contenido que tenga.

Ancho

La propiedad CSS que controla la anchura de la caja de los elementos se denomina `width`.

La propiedad `width` no admite valores negativos y los valores en porcentaje se calculan a partir de la anchura de su elemento padre. El valor `inherit` indica que la anchura del elemento se hereda de su elemento padre.

El siguiente ejemplo establece como valor de ancho 200px para el elemento `<div>` caja:

```
#caja { width: 200px; }

<div id="caja">
  ...
</div>
```

Otras dos propiedades relacionadas con la anchura de los elementos: `min-width` y `max-width`.

La propiedad `max-width` define el ancho de una caja, sin darle un valor fijo. De esta manera le podemos dar un tamaño de ancho máximo, lo que significa que si la ventana del navegador se achica o se agranda, la caja también lo hace, pero con un ancho máximo de tope. O sea, no se va a agrandar más que el valor que le hayamos configurado en la propiedad `max-width`.

```
#caja { width: 200px; max-width: 800px; }

<div id="caja">
  ...
</div>
```

De la misma manera que definimos un ancho máximo, podemos definir un ancho mínimo. Lo que significa que la caja no va a medir menos de lo que nosotros le hayamos configurado. Esta propiedad se llama `min-width`.

```
#caja { width: 200px; max-width: 800px; min-width: 400px; }

<div id="caja">
  ...
</div>
```

Para resumir, basándonos en el ejemplo anterior, nuestra caja no va a medir menos de 400px ni más de 800px, dentro de esas medidas y dependiendo del tamaño de la ventana del navegador, nuestra caja es variable.

Altura

Al igual que sucede con `width`, la propiedad `height` no admite valores negativos. Si se indica un porcentaje, se toma como referencia la altura del elemento padre. Si el elemento padre no tiene una altura definida explícitamente, se asigna el valor auto a la altura.

El siguiente ejemplo establece como valor de alto 50px para el elemento `<div>` cabecera:

```
#cabecera { height: 50px; }

<div id="cabecera">
  ...
</div>
```

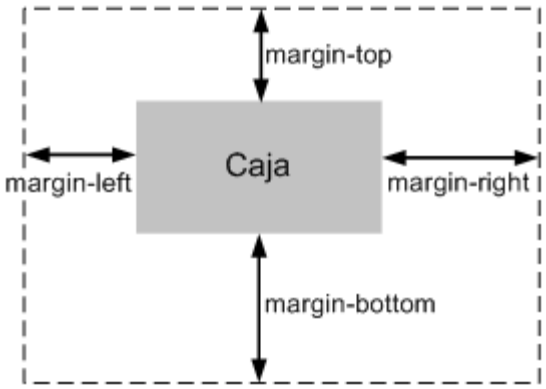
Otras dos propiedades relacionadas con la altura de los elementos: `min-height` y `max-height`.

Estas dos propiedad funcionan exáctamente igual que con el ancho. Nos permite definir topes máximos y mínimos en la altura de nuestra caja.

Márgenes

CSS define cuatro propiedades para controlar cada uno de los márgenes horizontales y verticales de un elemento.

Cada una de las propiedades establece la separación entre el borde lateral de la caja y el resto de cajas adyacentes:



Normalmente, las unidades más utilizadas para indicar los márgenes de un elemento son los píxeles.

Las propiedades `margin-top`, `margin-right`, `margin-bottom`, `margin-left` se utilizan para definir el ancho de los márgenes de cada uno de los lados del elemento por separado.

Puedes definir los 4 lados o solo aquellos que necesites.

En base a nuestro ejemplo anterior, vamos a definirle márgenes a nuestro div caja.

```
#caja { width: 200px; max-width: 800px; min-width: 400px; margin-top: 5px; margin-bottom: 5px;
 margin-left: 10px; margin-right: 10px; }
```

Además de las cuatro propiedades que controlan cada uno de los márgenes del elemento, CSS define una propiedad especial que permite establecer los cuatro márgenes de forma simultánea.

Esta propiedad se llama `margin`. Entonces:

1) Si sólo se indica un valor, todos los márgenes tienen ese mismo valor.

Ejemplo: `margin: 5px;`

En este caso, nuestra caja va a tener un margen de 5px en sus cuatro lados.

2) Si se indican dos valores, el primero se asigna al margen superior e inferior y el segundo se asigna a los márgenes izquierdo y derecho.

Ejemplo: `margin: 5px 10px;`

En este caso, nuestra caja va a tener un margen de 5px de arriba y de abajo y 10 px de la derecha y de la izquierda.

3) Si se indican los cuatro valores, el orden de asignación es: margen superior, margen derecho, margen inferior y margen izquierdo.

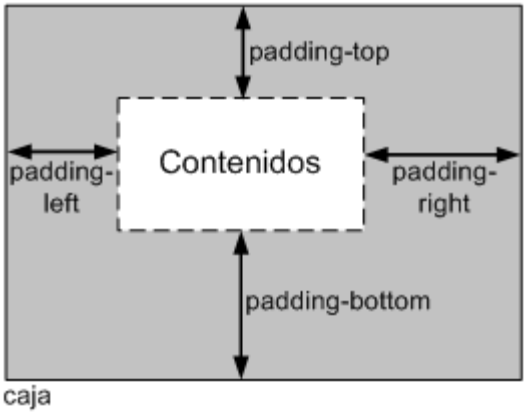
Ejemplo: `margin: 5px 10px 12px 15px;`

En este caso, nuestra caja va a tener un margen de 5px de arriba, 10px de la derecha, 12px de abajo y 15px de la izquierda.

Relleno

CSS define cuatro propiedades para controlar cada uno de los espacios de relleno horizontales y verticales de un elemento.

Cada una de estas propiedades establece la separación entre el contenido y los bordes laterales de la caja del elemento:



Al igual que con los márgenes, CSS también define una propiedad llamada `padding` para establecer los cuatro rellenos de un elemento de forma simultánea.

Entonces, podemos escribir cada relleno por separado, definiendo su ubicación:

```
padding-top: 5px;

padding-bottom: 5px;

padding-left: 10px;

padding-right: 10px;
```

O podemos escribir los cuatro juntos con la propiedad `padding` por sí sola:

```
padding: 5px 10px 12px 15px;
```

Flexbox

El objetivo de FlexBox es crear un modelo de caja o contenedor optimizado para los distintos dispositivos.

Lo que caracteriza un diseño flexible es su habilidad para alterar el ancho y alto de sus elementos para ajustarse lo mejor posible al espacio disponible en cualquier dispositivo. Un contenedor flexible expande sus elementos para rellenar el espacio libre, o los comprime para evitar que rebasen el área prevista.

Sintaxis

Para comenzar a trabajar con las propiedades Flexbox, primero tenemos que establecer el contenedor Flexible o “Flex Container”, para que todos los contenidos interiores, se comporten flexiblemente.

Esto lo hacemos con la propiedad display y el valor Flex. `display: flex;`

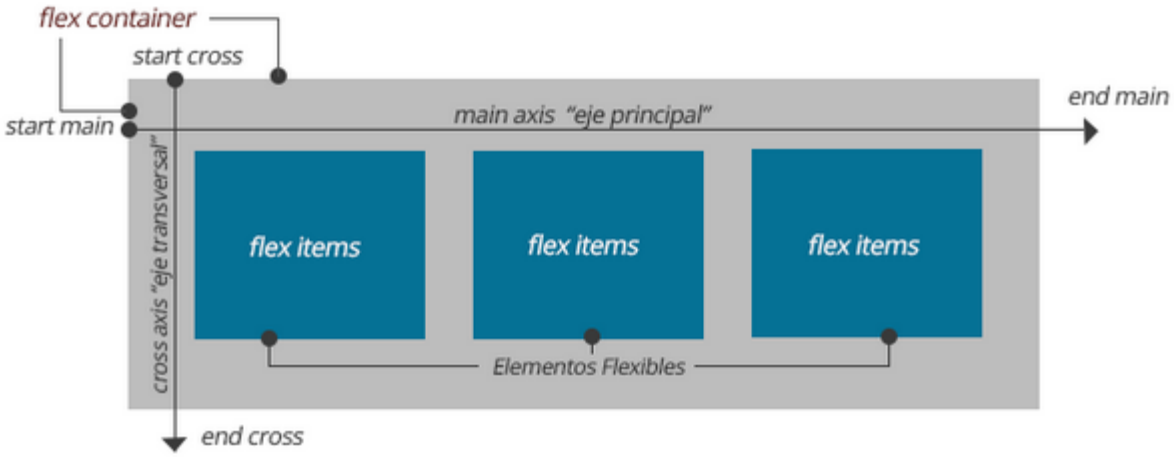
Para que podamos entender mejor, veamos la siguiente imagen:

Flex container: Es el contenedor principal flexible

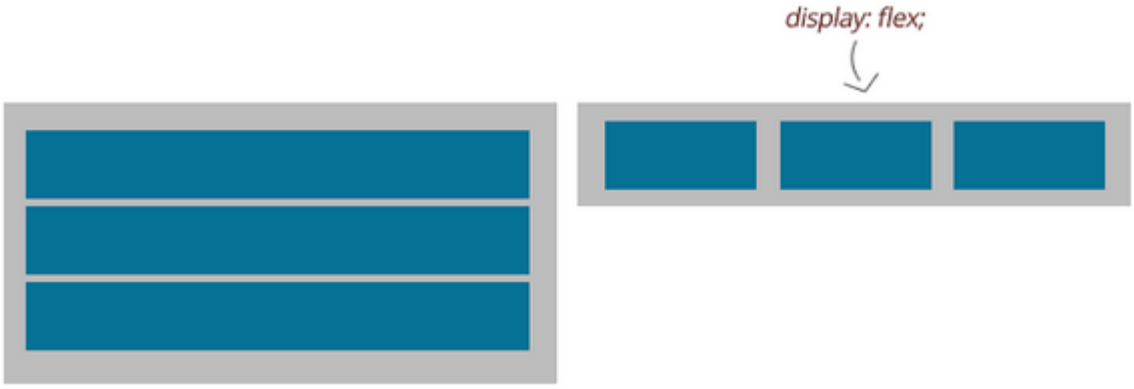
Flex items: Son los hijos directos del flex container y automáticamente se vuelven items flexibles.

Main axis: Es el eje principal que atraviesa el elemento flex container horizontalmente y va de izquierda a derecha en su flujo normal comenzando con start y terminando en el end main.

Cross axis: Es el eje vertical que atraviesa el elemento,



el eje transversal y también consta de un start y end, va de arriba a abajo del elemento.



Propiedades

Flex-direction: Con flex-direction, vamos a poder establecer la dirección de los flex ítems en el eje principal dentro del flex container. Podemos cambiar su orden, colocarlos en disposición de columnas e invertirlos en cualquier eje.

Flex-wrap: A veces cuando utilizamos las características Flexbox de CSS3 y el contenido es ejecutado en un viewport pequeño el contenido que es flexible tratara de ajustarse y algunas veces este tendera a salirse de su contendor principal, de su flex container.

Con **flex-wrap** vamos a controlar esta situación, si el contenido es mucho para mostrarse en una sola línea, en el eje principal, esta propiedad utilizara el eje transversal para reordenar los ítems flexibles.

Justify-content: Esta propiedad nos va a permitir controlar la alineación de los elementos flexibles en el main axis o eje principal.

Posicionamiento y visualización

Cuando los navegadores descargan el contenido HTML y CSS de las páginas web crean una caja para representar a cada elemento.
Los factores que se tienen en cuenta para generar cada caja son:

- Las propiedades `width`y `height` de la caja (si están establecidas).
- El tipo de cada elemento HTML (elemento de bloque o elemento en línea).
- Posicionamiento de la caja (normal, relativo, absoluto, fijo o flotante).
- Las relaciones entre elementos (dónde se encuentra cada elemento, elementos descendientes, etc.)
- Otro tipo de información, como por ejemplo el tamaño de las imágenes y el tamaño de la ventana del navegador.

Tipos de elementos

El estándar HTML clasifica a todos sus elementos en dos grandes grupos: elementos en línea (*inline*) y elementos de bloque (*block*).

Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea.
Por otra parte, los elementos en línea no empiezan necesariamente en nueva línea y sólo ocupan el espacio necesario para mostrar sus contenidos.

Los párrafos son elementos de bloque.

Los enlaces son elmentos de línea.

Dentro de un párrafo los enlaces siguen siendo elementos de línea.

Los elementos en línea definidos por HTML son:

`a`, `abbr`, `acronym`, `b`, `basefont`, `bdo`, `big`, `br`, `cite`, `code`, `dfn`, `em`, `font`, `i`, `img`, `input`, `kbd`, `label`, `q`, `s`,`samp`, `select`, `small`, `span`, `strike`, `strong`, `sub`, `sup`, `textarea`, `tt`, `u`, `var`.

Los elementos de bloque definidos por HTML son:

`address`, `blockquote`, `center`, `dir`, `div`, `dl`, `fieldset`, `form`, `h1`, `h2`,`h3`, `h4`, `h5`, `h6`, `hr`, `isindex`, `menu`, `noframes`, `noscript`, `ol`, `p`, `pre`, `table`, `ul`.

Los siguientes elementos también se considera que son de bloque:

`dd`, `dt`, `frameset`, `li`, `tbody`, `td`, `tfoot`, `th`, `thead`, `tr`.

Los siguientes elementos pueden ser en línea y de bloque según las circunstancias:

`button`, `del`, `iframe`, `ins`, `map`, `object`, `script`.

Posicionamiento

Los navegadores crean y posicionan de forma automática todas las cajas que forman cada página HTML. No obstante, CSS permite modificar la posición en la que se muestra cada caja.

Utilizando las propiedades que proporciona CSS para alterar la posición de las cajas es posible realizar efectos muy avanzados y diseñar estructuras de páginas que de otra forma no serían posibles.

El estándar de CSS define cinco modelos diferentes para posicionar una caja:

- Posicionamiento normal o estático: Se trata del posicionamiento que utilizan los navegadores si no se indica lo contrario.
- Posicionamiento relativo: Variante del posicionamiento normal que consiste en posicionar una caja según el posicionamiento normal y después desplazarla respecto de su posición original.
- Posicionamiento absoluto: La posición de una caja se establece de forma absoluta respecto de su elemento contenedor y el resto de elementos de la página ignoran la nueva posición del elemento.
- Posicionamiento fijo: Variante del posicionamiento absoluto que convierte una caja en un elemento inamovible, de forma que su posición en la pantalla siempre es la misma independientemente del resto de elementos e independientemente de si el usuario sube o baja la página en la ventana del navegador.
- Posicionamiento flotante: Se trata del modelo más especial de posicionamiento, ya que desplaza las cajas todo lo posible hacia la izquierda o hacia la derecha de la línea en la que se encuentran.

El posicionamiento de una caja se establece mediante la propiedad `position`. El significado de cada uno de los posibles valores de la propiedad es el siguiente:

- **static**: Corresponde al posicionamiento normal o estático. Si se utiliza este valor, se ignoran los valores de las propiedades top, right, bottom y left que se verán a continuación.
- **relative**: Corresponde al posicionamiento relativo. El desplazamiento de la caja se controla con las propiedades top, right, bottom y left.
- **absolute**: Corresponde al posicionamiento absoluto. El desplazamiento de la caja también se controla con las propiedades top, right, bottom y left, pero el origen de coordenadas del desplazamiento depende del posicionamiento de su elemento contenedor.
- **fixed**: Corresponde al posicionamiento fijo. El desplazamiento se establece de la misma forma que en el posicionamiento absoluto, pero en este caso el elemento permanece inamovible en la pantalla.

La propiedad `position` no permite controlar el posicionamiento flotante, que se establece con otra propiedad llamada `float` y que se explica más adelante. Además, la propiedad `position` sólo indica cómo se posiciona una caja, pero no la desplaza.

Normalmente, cuando se posiciona una caja también es necesario desplazarla respecto de su posición original o respecto de otro origen de coordenadas. CSS define cuatro propiedades llamadas `top`, `right`, `bottom` y `left` para controlar el desplazamiento de las cajas posicionadas.

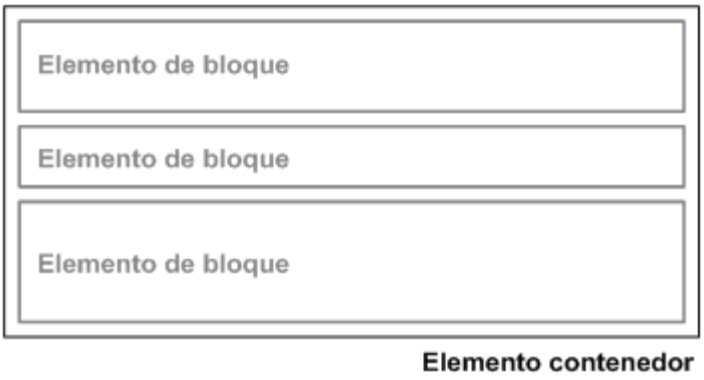
En el caso del posicionamiento relativo, cada una de estas propiedades indica el desplazamiento del elemento desde la posición original de su borde superior/derecho/inferior/izquierdo. Si el posicionamiento es absoluto, las propiedades indican el desplazamiento del elemento respecto del borde superior/derecho/inferior/izquierdo de su primer elemento padre posicionado.

En cualquiera de los dos casos, si el desplazamiento se indica en forma de porcentaje, se refiere al porcentaje sobre el ancho (propiedades `right` y `left`) o altura (propiedades `top` y `bottom`) del elemento.

Posicionamiento normal

El posicionamiento normal o estático es el modelo que utilizan por defecto los navegadores para mostrar los elementos de las páginas. En este modelo, sólo se tiene en cuenta si el elemento es de bloque o en línea, sus propiedades `width` y `height` y su contenido.

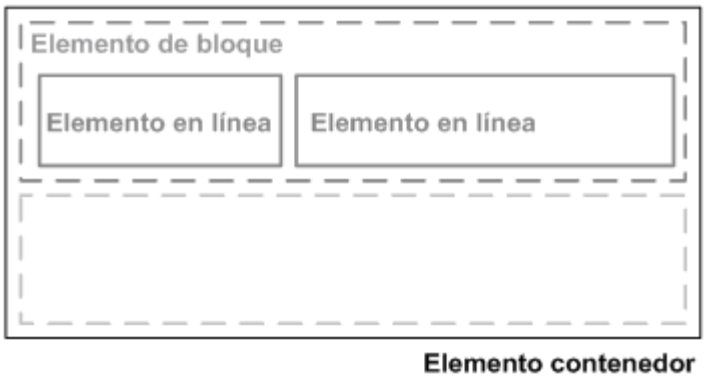
Los elementos de bloque forman lo que CSS denomina *"contextos de formato de bloque"*. En este tipo de contextos, las cajas se muestran una debajo de otra comenzando desde el principio del elemento contenedor. La distancia entre las cajas se controla mediante los márgenes verticales.



Si un elemento se encuentra dentro de otro, el elemento padre se llama *"elemento contenedor"* y determina tanto la posición como el tamaño de todas sus cajas interiores.

Si un elemento no se encuentra dentro de un elemento contenedor, entonces su elemento contenedor es el elemento `<body>` de la página. Normalmente, la anchura de los elementos de bloque está limitada a la anchura de su elemento contenedor, aunque en algunos casos sus contenidos pueden desbordar el espacio disponible.

Los elementos en línea forman los *"contextos de formato en línea"*. En este tipo de contextos, las cajas se muestran una detrás de otra de forma horizontal comenzando desde la posición más a la izquierda de su elemento contenedor. La distancia entre las cajas se controla mediante los márgenes laterales.



Si las cajas en línea ocupan más espacio del disponible en su propia línea, el resto de cajas se muestran en las líneas inferiores. Si las cajas en línea ocupan un espacio menor que su propia línea, se puede controlar la distribución de las cajas mediante la propiedad `text-align` para centrarlas, alinearlas a la derecha o justificarlas.

Posicionamiento relativo

El estándar CSS considera que el posicionamiento relativo es un caso particular del posicionamiento normal, aunque en

la práctica presenta muchas diferencias.

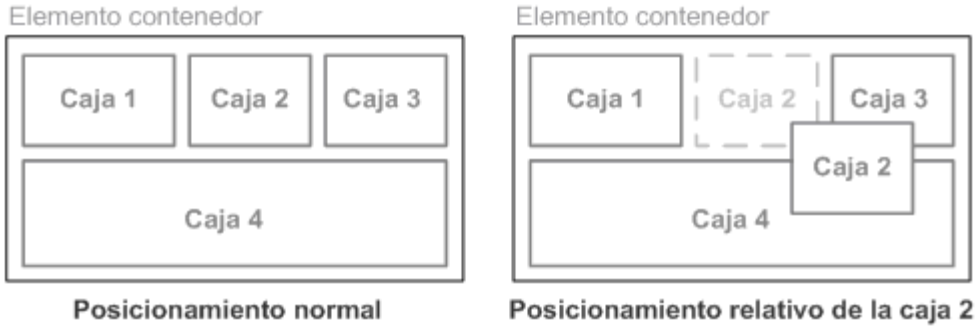
El posicionamiento relativo desplaza una caja respecto de su posición original establecida mediante el posicionamiento normal. El desplazamiento de la caja se controla con las propiedades `top`, `right`, `bottom` y `left`.

El valor de la propiedad `top` se interpreta como el desplazamiento entre el borde superior de la caja en su posición final y el borde superior de la misma caja en su posición original.

De la misma forma, el valor de las propiedades `left`, `right` y`bottom` indica respectivamente el desplazamiento entre el borde izquierdo/derecho/inferior de la caja en su posición final y el borde izquierdo/derecho/inferior de la caja original.

Por tanto, la propiedad `top` se emplea para mover las cajas de forma descendente, la propiedad `bottom` mueve las cajas de forma ascendente, la propiedad `left` se utiliza para desplazar las cajas hacia la derecha y la propiedad `right` mueve las cajas hacia la izquierda. Este comportamiento parece poco intuitivo y es causa de errores cuando se empiezan a diseñar páginas con CSS. Si se utilizan valores negativos en las propiedades `top`,`right`,`bottom` y`left`, su efecto es justamente el inverso.

El desplazamiento relativo de una caja no afecta al resto de cajas adyacentes, que se muestran en la misma posición que si la caja desplazada no se hubiera movido de su posición original.



En la imagen anterior, la caja 2 se ha desplazado lateralmente hacia la derecha y verticalmente de forma descendente. Como el resto de cajas de la página no modifican su posición, se producen solapamientos entre los contenidos de las cajas.

Las cajas desplazadas de forma relativa no modifican su tamaño, por lo que los valores de las propiedades `left` y `right` siempre cumplen que `left = -right`.

Si tanto `left` como `right` tienen un valor de `auto` (que es su valor por defecto) la caja no se mueve de su posición original. Si sólo el valor de `left` es `auto`, su valor real es`-right`. Igualmente, si sólo el valor de `right` es `auto`, su valor real es`-left`.

Si tanto`left` como `right` tienen valores distintos de `auto`, uno de los dos valores se tiene que ignorar porque son mutuamente excluyentes. Para determinar la propiedad que se tiene en cuenta, se considera el valor de la propiedad `direction`.

La propiedad `direction` permite establecer la dirección del texto de un contenido. Si el valor de `direction` es`ltr`, el texto se muestra de izquierda a derecha, que es el método de escritura habitual en la mayoría de países. Si el valor de `direction` es `rtl`, el método de escritura es de derecha a izquierda.

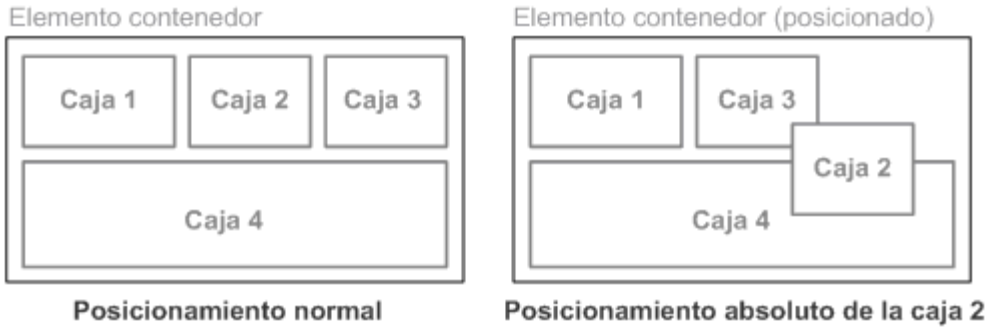
Si el valor de `direction` es `ltr`, y las propiedades `left` y `right` tienen valores distintos de `auto`, se ignora la propiedad `right` y sólo se tiene en cuenta el valor de la propiedad `left`. De la misma forma, si el valor de `direction` es `rtl`, se ignora el valor de `left` y sólo se tiene en cuenta el valor de `right`.

Posicionamiento absoluto

El posicionamiento absoluto se emplea para establecer de forma exacta la posición en la que se muestra la caja de un elemento. La nueva posición de la caja se indica mediante las propiedades `top`, `right`, `bottom` y `left`. La interpretación de los valores de estas propiedades es mucho más compleja que en el posicionamiento relativo, ya que en este caso dependen del posicionamiento del elemento contenedor.

Cuando una caja se posiciona de forma absoluta, el resto de elementos de la página se ven afectados y modifican su posición. Al igual que en el posicionamiento relativo, cuando se posiciona de forma absoluta una caja es probable que se produzcan solapamientos con otras cajas.

En el siguiente ejemplo, se posiciona de forma absoluta la caja 2:



La caja 2 está posicionada de forma absoluta, lo que provoca que el resto de elementos de la página modifiquen su posición. En concreto, la caja 3 deja su lugar original y pasa a ocupar el hueco dejado por la caja 2.

Por otra parte, el desplazamiento de una caja posicionada de forma absoluta se controla mediante las propiedades `top`, `right`, `bottom` y `left`. A diferencia del posicionamiento relativo, la interpretación de los valores de estas propiedades depende del elemento contenedor de la caja posicionada.

Determinar la referencia utilizada para interpretar los valores de `top`, `right`, `bottom` y `left` de una caja posicionada de forma absoluta es un proceso complejo que se compone de los siguientes pasos:

- Se buscan todos los elementos contenedores de la caja hasta llegar al elemento `<body>` de la página.
- Se recorren todos los elementos contenedores empezando por el más cercano a la caja y llegando hasta el `<body>`
- El primer elemento contenedor que esté posicionado de cualquier forma diferente a `position: static` se convierte en la referencia que determina la posición de la caja posicionada de forma absoluta.
- Si ningún elemento contenedor está posicionado, la referencia es la ventana del navegador, que no debe confundirse con el elemento `<body>` de la página.

Una vez determinada la referencia del posicionamiento absoluto, la interpretación de los valores de las propiedades `top`, `right`, `bottom` y `left` se realiza como sigue:

- El valor de la propiedad `top` indica el desplazamiento desde el borde superior de la caja hasta el borde superior del elemento contenedor que se utiliza como referencia.
- El valor de la propiedad `right` indica el desplazamiento desde el borde derecho de la caja hasta el borde derecho del elemento contenedor que se utiliza como referencia.
- El valor de la propiedad `bottom` indica el desplazamiento desde el borde inferior de la caja hasta el borde inferior del elemento contenedor que se utiliza como referencia.
- El valor de la propiedad `left` indica el desplazamiento desde el borde izquierdo de la caja hasta el borde izquierdo del elemento contenedor que se utiliza como referencia.

Posicionamiento fijo

El estándar CSS considera que el posicionamiento fijo es un caso particular del posicionamiento absoluto, ya que sólo se diferencian en el comportamiento de las cajas posicionadas.

Cuando una caja se posiciona de forma fija, la forma de obtener el origen de coordenadas para interpretar su desplazamiento es idéntica al posicionamiento absoluto. De hecho, si el usuario no mueve la página HTML en la ventana del navegador, no existe ninguna diferencia entre estos dos modelos de posicionamiento.

La principal característica de una caja posicionada de forma fija es que su posición es inamovible dentro de la ventana del navegador. El posicionamiento fijo hace que las cajas no modifiquen su posición ni aunque el usuario suba o baje la página en la ventana de su navegador.

Si la página se visualiza en un medio paginado (por ejemplo en una impresora) las cajas posicionadas de forma fija se repiten en todas las páginas. Esta característica puede ser útil para crear encabezados o pies de página en páginas HTML preparadas para imprimir.

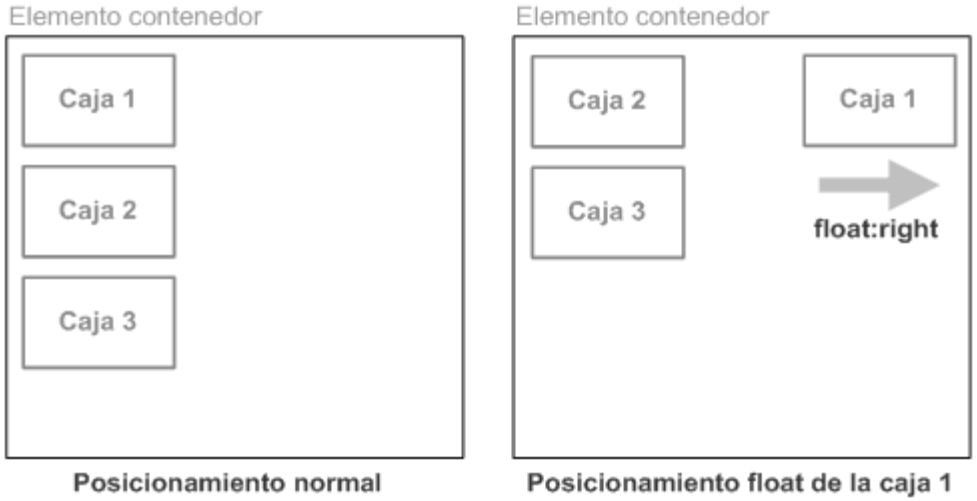
El posicionamiento fijo apenas se ha utilizado en el diseño de páginas web hasta hace poco tiempo porque el navegador Internet Explorer 6 y las versiones anteriores no lo soportan.

Posicionamiento flotante

El posicionamiento flotante es el más difícil de comprender pero al mismo tiempo es el más utilizado. La mayoría de estructuras de las páginas web complejas están diseñadas con el posicionamiento flotante, como se verá más adelante.

Cuando una caja se posiciona con el modelo de posicionamiento flotante, automáticamente se convierte en una *caja flotante*, lo que significa que se desplaza hasta la zona más a la izquierda o más a la derecha de la posición en la que originalmente se encontraba.

La siguiente imagen muestra el resultado de posicionar de forma flotante hacia la derecha la caja 1:



Cuando se posiciona una caja de forma flotante:

- La caja deja de pertenecer al flujo normal de la página, lo que significa que el resto de cajas ocupan el lugar

dejado por la caja flotante.
- La caja flotante se posiciona lo más a la izquierda o lo más a la derecha posible de la posición en la que se encontraba originalmente.

Las cajas flotantes influyen en la disposición de todas las demás cajas. Los elementos en línea hacen sitio a las cajas flotantes adaptando su anchura al espacio libre dejado por la caja desplazada. Los elementos de bloque no les hacen sitio, pero sí que adaptan sus contenidos para que no se solapen con las cajas flotantes.

La propiedad CSS que permite posicionar de forma flotante una caja se denomina `float`:

Si se indica un valor `left`, la caja se desplaza hasta el punto más a la izquierda posible en esa misma línea (si no existe sitio en esa línea, la caja baja una línea y se muestra lo más a la izquierda posible en esa nueva línea). El resto de elementos adyacentes se adaptan y fluyen alrededor de la caja flotante.

El valor `right` tiene un funcionamiento idéntico, salvo que en este caso, la caja se desplaza hacia la derecha. El valor `none` permite anular el posicionamiento flotante de forma que el elemento se muestre en su posición original.

La propiedad `clear` permite modificar el comportamiento por defecto del posicionamiento flotante para forzar a un elemento a mostrarse debajo de cualquier caja flotante. La regla CSS que se aplica al segundo párrafo del ejemplo anterior es la siguiente:

```
<p style="clear: left;">...</p>
```

La propiedad `clear` indica el lado del elemento HTML que no debe ser adyacente a ninguna caja posicionada de forma flotante. Si se indica el valor `left`, el elemento se desplaza de forma descendente hasta que pueda colocarse en una línea en la que no haya ninguna caja flotante en el lado izquierdo.

La especificación oficial de CSS explica este comportamiento como *"un desplazamiento descendente hasta que el borde superior del elemento esté por debajo del borde inferior de cualquier elemento flotante hacia la izquierda"*.

Si se indica el valor `right`, el comportamiento es análogo, salvo que en este caso se tienen en cuenta los elementos desplazados hacia la derecha.

El valor `both` despeja los lados izquierdo y derecho del elemento, ya que desplaza el elemento de forma descendente hasta que el borde superior se encuentre por debajo del borde inferior de cualquier elemento flotante hacia la izquierda o hacia la derecha.

Visualización

Además de las propiedades que controlan el posicionamiento de los elementos, CSS define otras cuatro propiedades para controlar su visualización: `display`, `visibility`, `overflow` y `z-index`.

Utilizando algunas de estas propiedades es posible ocultar y/o hacer invisibles las cajas de los elementos, por lo que son imprescindibles para realizar efectos avanzados y animaciones.

Propiedades display y visibility

Las propiedades `display` y `visibility` controlan la visualización de los elementos. Las dos propiedades permiten ocultar cualquier elemento de la página. Habitualmente se utilizan junto con JavaScript para crear efectos dinámicos como mostrar y ocultar determinados textos o imágenes cuando el usuario pincha sobre ellos.

La propiedad `display` permite ocultar completamente un elemento haciendo que desaparezca de la página. Como el

elemento oculto no se muestra, el resto de elementos de la página se mueven para ocupar su lugar.

Por otra parte, la propiedad `visibility` permite hacer invisible un elemento, lo que significa que el navegador crea la caja del elemento pero no la muestra. En este caso, el resto de elementos de la página no modifican su posición, ya que aunque la caja no se ve, sigue ocupando sitio.

En general, cuando se oculta un elemento no es deseable que siga ocupando sitio en la página, por lo que la propiedad `display` se utiliza mucho más que la propiedad `visibility`.

Las posibilidades de la propiedad `display` son mucho más avanzadas que simplemente ocultar elementos. En realidad, la propiedad `display` modifica la forma en la que se visualiza un elemento.

Los valores más utilizados son `inline`, `block` y `none`. El valor `block` muestra un elemento como si fuera un elemento de bloque, independientemente del tipo de elemento que se trate. El valor `inline` visualiza un elemento en forma de elemento en línea, independientemente del tipo de elemento que se trate.

El valor `none` oculta un elemento y hace que desaparezca de la página. El resto de elementos de la página se visualizan como si no existiera el elemento oculto, es decir, pueden ocupar el espacio en el que se debería visualizar el elemento.

Todos los valores de la propiedad `display` son: `inline`, `block`, `none`, `list-item`, `run-in`, `inline-block`, `table`, `inline-table`, `table-row-group`, `table-header-group`, `table-footer-group`, `table-row`, `table-column-group`, `table-column`, `table-cell`, `table-caption`, `inherit`.

La propiedad `display: inline`, se puede utilizar en las listas (`<ul>`, `<ol>`) que se quieren mostrar horizontalmente y la propiedad `display: block`, se emplea frecuentemente para los enlaces que forman el menú de navegación.

Las posibilidades de la propiedad `visibility`, son mucho más limitadas que las de la propiedad `display`, ya que sólo permite hacer visibles o invisibles a los elementos de la página.

Inicialmente todas las cajas que componen la página son visibles. Empleando el valor `hidden`, es posible convertir una caja en invisible para que no muestre sus contenidos. El resto de elementos de la página se muestran como si la caja todavía fuera visible, por lo que en el lugar donde originalmente se mostraba la caja invisible, ahora se muestra un hueco vacío.

Por último, el valor `collapse`, de la propiedad `visibility`, sólo se puede utilizar en las filas, grupos de filas, columnas y grupos de columnas de una tabla. Su efecto es similar al de la propiedad `display`, ya que oculta completamente la fila y/o columna y se pueden mostrar otros contenidos en ese lugar. Si se utiliza el valor collapse sobre cualquier otro tipo de elemento, su efecto es idéntico al valor `hidden`.

Relación entre display, float y position

Cuando se establecen las propiedades `display`, `float` y `position`, sobre una misma caja, su interpretación es la siguiente:

- Si `display` vale `none`, se ignoran las propiedades `float` y `position`, y la caja no se muestra en la página.
- Si `position` vale `absolute` o `fixed`, la caja se posiciona de forma absoluta, se considera que `float` vale `none` y la propiedad `display` vale `block` tanto para los elementos en línea como para los elementos de bloque. La posición de la caja se determina mediante el valor de las propiedades `top`, `right`, `bottom` y `left`.
- En cualquier otro caso, si `float` tiene un valor distinto de `none`, la caja se posiciona de forma flotante y la propiedad `display` vale `block` tanto para los elementos en línea como para los elementos de bloque.

Propiedad overflow

Normalmente, los contenidos de un elemento se pueden mostrar en el espacio reservado para ese elemento. Sin

embargo, en algunas ocasiones el contenido de un elemento no cabe en el espacio reservado para ese elemento y se desborda.

La situación más habitual en la que el contenido sobresale de su espacio reservado es cuando se establece la anchura y/o altura de un elemento mediante la propiedad `width` y/o `height`. Otra situación habitual es la de las líneas muy largas contenidas dentro de un elemento `<pre>`, que hacen que la página entera sea demasiado ancha.

CSS define la propiedad `overflow` para controlar la forma en la que se visualizan los contenidos que sobresalen de sus elementos.

Los valores de la propiedad overflow tienen el siguiente significado:

- `visible`: el contenido no se corta y se muestra sobresaliendo la zona reservada para visualizar el elemento. Este es el comportamiento por defecto.
- `hidden`: el contenido sobrante se oculta y sólo se visualiza la parte del contenido que cabe dentro de la zona reservada para el elemento.
- `scroll`: solamente se visualiza el contenido que cabe dentro de la zona reservada para el elemento, pero también se muestran barras de scroll que permiten visualizar el resto del contenido.
- `auto`: el comportamiento depende del navegador, aunque normalmente es el mismo que la propiedad `scroll`.

Propiedad z-index

Además de posicionar una caja de forma horizontal y vertical, CSS permite controlar la posición tridimensional de las cajas posicionadas. De esta forma, es posible indicar las cajas que se muestran delante o detrás de otras cajas cuando se producen solapamientos.

La posición tridimensional de un elemento se establece sobre un tercer eje llamado Z y se controla mediante la propiedad `z-index`. Utilizando esta propiedad es posible crear páginas complejas con varios niveles o capas.

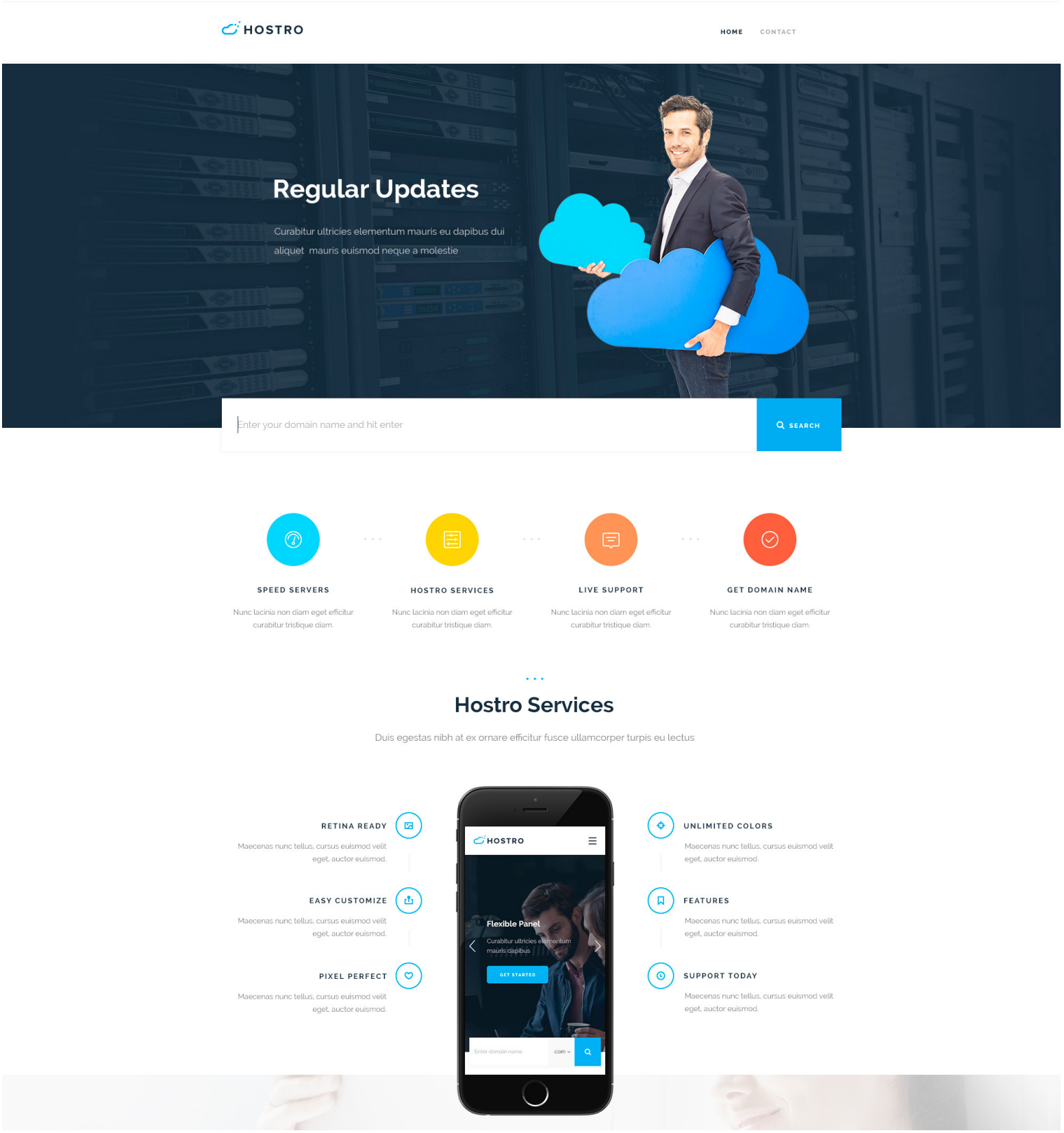
El valor más común de la propiedad `z-index` es un número entero. Aunque la especificación oficial permite los números negativos, en general se considera el número 0 como el nivel más bajo.

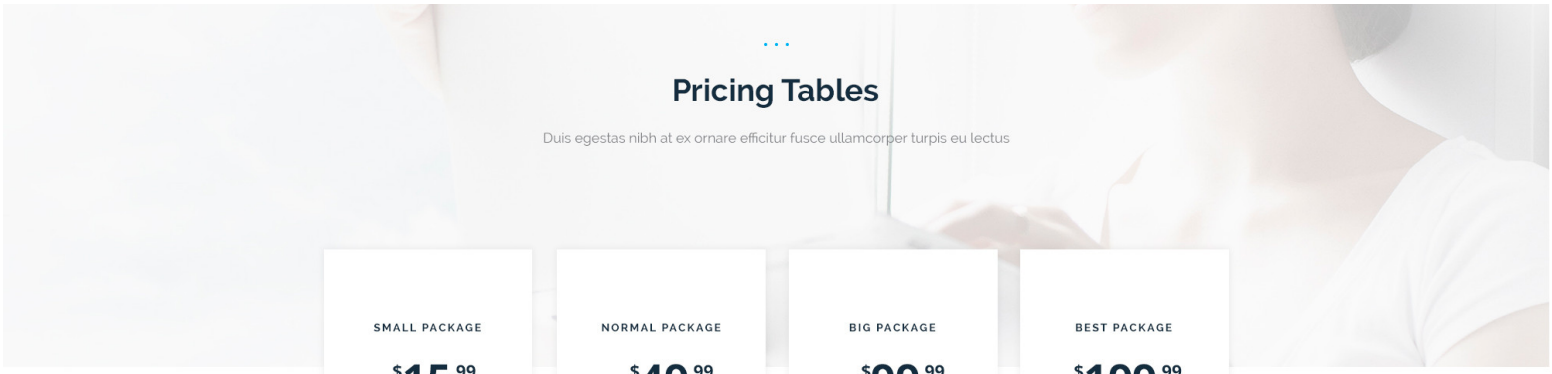
Cuanto más alto sea el valor numérico, más cerca del usuario se muestra la caja. Un elemento con `z-index`: 10 se muestra por encima de los elementos con `z-index: 8` o `z-index: 9`, pero por debajo de elementos con `z-index: 20` o `z-index: 50`.

Trabajo Práctico 4: CSS

Descripción:

Posicionar los elementos como aparecen en el diseño.





Pricing Tables

Duis egestas nibh at ex ornare efficitur fusce ullamcorper turpis eu lectus

SMALL PACKAGE

$15.99
/mount

Hard drive, GB	2
Transfer, GB	1
Data bases	1
Dashboards	1

SELECT PLAN

NORMAL PACKAGE

$49.99
/mount

Hard drive, GB	5
Transfer, GB	2
Data bases	2
Dashboards	2

SELECT PLAN

BIG PACKAGE

$99.99
/mount

Hard drive, GB	10
Transfer, GB	3
Data bases	3
Dashboards	3

SELECT PLAN

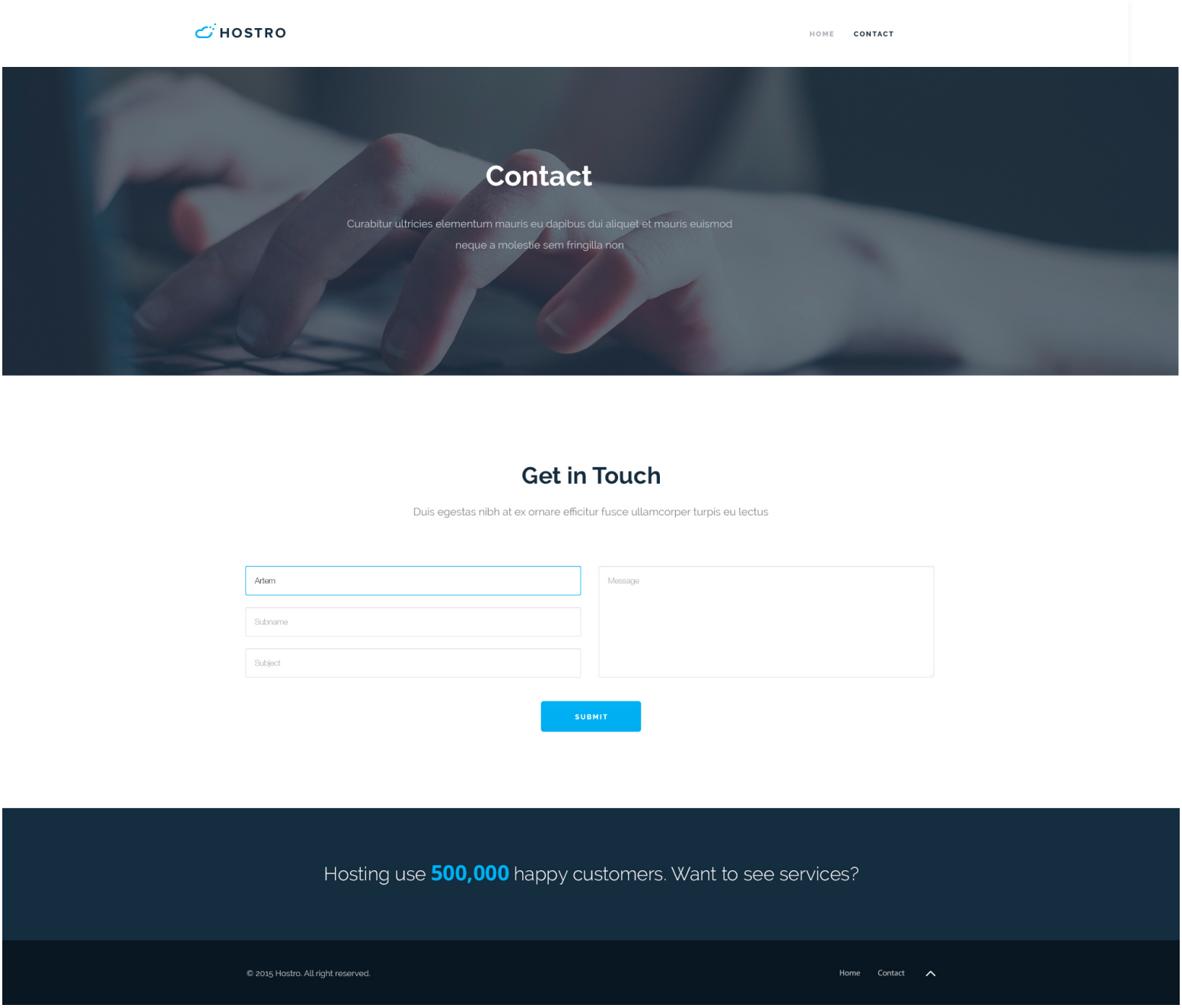
BEST PACKAGE

$199.99
/mount

Hard drive, GB	15
Transfer, GB	4
Data bases	4
Dashboards	4

SELECT PLAN

Hosting use **500,000** happy customers. Want to see services?



Objetivo:

- Entender los distintos modelos de posicionamiento(relativo, absoluto, flexbox, fixed, float)

Requerimientos:

- Haber realizado los trabajos anteriores
- Comprender cómo se comportan los modelos(block, * inline-block, inline).
- Subir la resolución de tu trabajo a tu repositorio de Github en la subcarpeta “CSS TP4”.

Bonus:

- Que es grid layout, es soportado por los navegadores?

Recursos:

- Tutorial de flexboxs
- Modelo de cajas
- Uso de float